

50325-0801 (Seq. No. 7693)

Patent

UNITED STATES PATENT APPLICATION

FOR

METHOD AND APPARATUS PROVIDING ROLE-BASED CONFIGURATION OF A PORT OF A
NETWORK ELEMENT

INVENTORS:

MICHAEL RIGGINS
STEVEN BERL
DIRK ARTEUNIS
ARNOLD STAMLER

PREPARED BY:

HICKMAN PALERMO TRUONG & BECKER LLP
1600 WILLOW STREET
SAN JOSE, CA 95125
(408) 414-1080

EXPRESS MAIL MAILING INFORMATION

"Express Mail" mailing label number: EV323354937

Date of Deposit: October 7, 2003

METHOD AND APPARATUS PROVIDING ROLE-BASED CONFIGURATION OF A PORT OF A NETWORK ELEMENT

FIELD OF THE INVENTION

[0001] The present invention generally relates to network management. The invention relates more specifically to a method and apparatus providing role-based configuration of a port of a network element.

COPYRIGHT NOTICE

[0002] A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction of the patent disclosure, as it appears in the Patent & Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever. Copyright © 2003 Cisco Systems, Inc.

BACKGROUND OF THE INVENTION

[0003] The approaches described in this section could be pursued, but are not necessarily approaches that have been previously conceived or pursued. Therefore, unless otherwise indicated herein, the approaches described in this section are not prior art to the claims in this application and are not admitted to be prior art by inclusion in this section.

[0004] In computer networks, network elements such as routers, switches and others are responsible for moving packetized information within the network and directing the information to end station devices. Information arriving at or sent from a network element is communicated using one or more ports. A port may have a direct physical association with a

chassis connector that terminates a network cable. Additionally or alternatively, port(s) may have a logical association with a connector.

[0005] The role that a port plays in a network element, or the identity of the neighbor of the port, can dictate a specific configuration for that port. For example, the configuration of a switch port that is connected to a PC might be very different than the configuration of the same port when connected to a business-critical server. As another hypothetical example, the configuration of the port might be different if a WAN link to a distant router is attached as opposed to a lab router connected via a LAN. To configure the port correctly, network administrators need to know the role of a port or the identity of the device to which it is attached.

[0006] At present, acquiring such knowledge is a manual process. The manual process is time-consuming and requires manual record-keeping. Further, if an end station device is changed, a port connected to that device may require re-configuration to account for differences in the new end station device. Currently, this requires manual intervention by an administrator, which is costly.

[0007] The Cisco Networking Services (CNS) solution from Cisco Systems, Inc. can discover information about the physical setup of modular routers including what kind of ports are present; however, CNS does not determine information about which devices are connected to the ports that it has discovered.

[0008] In one related approach, an “Auto-Configuration” mechanism uses a Trivial File Transfer Protocol (TFTP) server to load a configuration to a network switch. The Cisco Intelligent Engine 2100 (IE 2100) uses a similar approach, but transfers configuration information using HTTP. However, both of these approaches impose configuration challenges relating to proper configuration of the associated servers, and impose higher costs

for servers on their users. Further, while the IE2100 can dynamically create a configuration for a device based on the types of ports discovered, it cannot do so based on what devices are connected to the ports.

[0009] In another approach, a network switch or router stores one or more role-based macro configuration templates. Each template is associated with a LAN switch role, such as “access,” “distribution,” or “core.” The templates apply particular configuration commands to interfaces of the switch depending on the associated role of the switch port. However, this approach does not discover devices that are connected to ports of the switch in the network, and does not determine what role is played by devices that are connected to ports of the switch. The network administrator is required to use a separate manual process to perform any such discovery and role determination, with the associated burden of keeping appropriate records and responding to changes manually as the changes occur. The approach also does not automatically apply a different configuration to each port based on determining the role of a connected device.

[0010] In another approach, RADIUS attribute-value pairs may be used to provision services on sessions based on the identity of a user; in this context, a session is defined by information specifying a port and a user for a period of time. For example, a user dials in to an access router, PPP authentication is performed via RADIUS, and the RADIUS reply contains configuration parameters for a port that are applied by the access router. When the connection terminates, a default configuration is restored to the port. However, this approach does not establish a port configuration based on the type of device coupled to the port or the role played by the port with respect to the device.

[0011] Thus, there is a need in this field for an improved automated way to configure ports of network elements based on discovering what devices are connected to the ports and what roles the ports or devices have in the network.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

[0013] FIG. 1 is a simplified block diagram of a network element configured to provide role-based configuration of a port;

[0014] FIG. 2 is a flow diagram that illustrates a high level overview of one embodiment of a method for role-based configuration of a port;

[0015] FIG. 3A is a flow diagram that illustrates another embodiment of a method for role-based configuration of a port;

[0016] FIG. 3B is a flow diagram showing further steps in the method of FIG. 3A;

[0017] FIG. 4 is a block diagram of example device port profile rules;

[0018] FIG. 5 is a block diagram of an example device port role mapping; and

[0019] FIG. 6 is a block diagram that illustrates a computer system upon which an embodiment may be implemented.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0020] A method and apparatus providing role-based configuration of a port of a network element is described. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form in order to avoid unnecessarily obscuring the present invention.

[0021] Embodiments are described herein according to the following outline:

- 1.0 General Overview
- 2.0 Structural and Functional Overview
 - 2.1 Architecture
 - 2.2 Role-Based Configuration Approaches
 - 2.3 Complete Example
 - 2.4 Time of Execution
 - 2.5 Interaction with Management Applications and Other Solutions
 - 2.6 Applicability to Physical Ports and Logical Ports
 - 2.7 Failure Processing
- 3.0 Implementation Mechanisms—Hardware Overview
- 4.0 Extensions and Alternatives

1.0 GENERAL OVERVIEW

[0022] The needs identified in the foregoing Background, and other needs and objects that will become apparent for the following description, are achieved in the present invention, which comprises, in one aspect, a method for automatically configuring a port of a network element, comprising the steps of discovering information that identifies or describes

a second network element that is coupled to a port of a first network element; associating the port with a port role definition selected from a plurality of port role definitions based on the discovered information; retrieving one or more configuration settings that are associated with the selected port role definition; and applying the one or more configuration settings to the port.

[0023] According to one feature of this aspect, the step of associating a port with a port role definition comprises the steps of mapping the port to a profile selected from a plurality of profiles based on the discovered information; and matching the profile to a port role definition. In another feature, a profile comprises a profile identifier; and one or more rules that maps one or more ports to the profile based on the discovered information. In another feature, a port role definition comprises a port role definition identifier, and one or more configuration values that can be applied to a port that is associated with the port role definition. In yet another feature, the configuration values comprise one or more configuration attributes and value pairs for the port.

[0024] According to yet another feature, the configuration values comprise one or more quality of service values for the port. In still another feature, the discovered information comprises a type identifier of the second element; and an identity identifier of the second element. In another feature, the type identifier of the second element is the capabilities TLV or platform TLV of a Cisco Discovery Protocol (CDP) message from the second element; the identity identifier of the second element is the sysName TLV of a CDP message from the second element.

[0025] In still another feature, the method further comprises providing the discovered information to a management application; receiving one or more configuration settings from the management application; applying the configuration settings to the port. In another

feature, the additional configuration settings are generated based on the discovered information. The first element may be programmed to automatically configure a selection of one or more ports from a plurality of ports. The steps may be performed when the first network element boots up, or the steps are performed for a particular port of the first network element when said port becomes active after a period of inactivity, or the steps are performed at a specified periodic time interval.

[0026] In another feature, the step of discovering information comprises receiving a Cisco Discovery Protocol (CDP) message from the second element, and extracting a Capabilities type-length-value (TLV) or platform TLV and a sysName TLV from the message. In another feature, the profile further comprises a counter indicating a number of ports to which the profile is currently mapped. The profile may further comprise a counter indicating a number of ports to which the profile is currently mapped, and the steps of retrieving and applying may be performed only when the counter is less than or equal to one.

[0027] In various other embodiments, the first network element is a switch or router; the second network element is a personal computer, phone, router, server, or storage device; and the step of discovering is performed by a protocol selected from the set consisting of Cisco Discovery Protocol, Network-Based Application Recognition, Netflow, SSG, 802.1x, and Simple Network Management Protocol. In an embodiment in which the step of discovering is performed by Simple Network Management Protocol, the discovered information comprises a sysDescr object and sysObjectID of a MIB of the second network element.

[0028] In other aspects, the invention encompasses a computer apparatus and a computer-readable medium configured to carry out the foregoing steps.

2.0 STRUCTURAL AND FUNCTIONAL OVERVIEW

[0029] According to one embodiment, a port of a network element automatically detects its role in a network based on the type or identity of a device to which the port is connected. A configuration of the port is set automatically to match the requirements of the device attached to that port. In one approach, a combination of network element intelligence and a management application provides an environment in which a network element makes an educated guess about the nature of the role of a port, or the identity of a neighbor. The network element uses that information, alone or in conjunction with other management applications, to automate the configuration of the port. In one embodiment, the management application is the Cisco CNS configuration engine, from Cisco Systems, Inc., San Jose, California.

2.1 ARCHITECTURE

[0030] FIG. 1 is a simplified block diagram of a network element configured to provide role-based configuration of a port. A first network element 102 comprises an operating system 104 having a discovery agent 106 and a role agent 108. In one embodiment, network element 102 is a switch, router, or other network infrastructure element, such as a 7200 series router from Cisco Systems, Inc. Operating system 104 may be the Cisco IOS® operating system.

[0031] First network element 102 also comprises one or more ports 110A, 110B, 110C that are communicatively coupled, directly or indirectly through one or more networks 114, to second network elements 112A, 112B, 112C, etc. Each of the second network elements 112A, 112B, 112C may be a switch, router, or other network infrastructure element, or an end station element such as a personal computer, workstation, server, printer, phone, etc.

[0032] Optionally, first network element 102 also is communicatively coupled to a management application 120 for purposes of configuring or managing the first network element. In other embodiments, the management application may execute within the first network element. For example, a personal computer running Linux can be configured to act as a router and may concurrently execute a network management application, or the Linux router PC may interoperate with a second PC that runs the network management application. Thus, the specific location of the management application or process is not critical to embodiments of the invention.

[0033] Role agent 108 can access one or more stored profile rules 130 and one or more role mappings 140. In one embodiment, profile rules 130 map discovered device identity information to profile names and role names, and each of the role mappings 140 maps a role name to one or more configuration settings, commands or values. Alternatively, role mappings 140 may contain the configuration settings, commands or values. Such configuration settings, commands or values may comprise any valid expression that conforms to a configuration language of a network element. FIG. 4 is an example of profile rules 130, and FIG. 5 is an example of role mappings 140. FIG. 4 and FIG. 5 are described further below.

[0034] Discovery agent 106 and role agent 108 each comprise one or more software elements, hardware elements, or a combination thereof that are configured to perform the functions that are described herein. Discovery agent 106 and role agent 108 cooperate with one another, and optionally with management application 120, to provide automated configuration of a port of the first network element as further described herein. Generally, discovery agent 106 discovers the particular kind or identity of device 112A, 112B, 112C that is coupled to the other end of each port or interface of a network element. Role agent 108

decides which role is assigned to different network elements. The following sections present specific approaches and examples of the preceding general principles.

2.2 ROLE-BASED CONFIGURATION APPROACHES

[0035] FIG. 2 is a flow diagram that illustrates a high level overview of one embodiment of a method for role-based configuration of a port.

[0036] In block 202, the method discovers information about a second network element that is connected to a port of a first network element. For example, discovery agent 106 of first network element 102 discovers information about second network element 112A, which is communicatively coupled to port 110A. In general, discovery agent 106 discovers role information and identity information for each second network element that is connected to each port or interfaces of network element 102. Discovery agent 106 may use any of a variety of protocols to deduce or derive such information.

[0037] In one embodiment, Cisco Discovery Protocol (CDP) is used for devices that support CDP; a detailed description of CDP is provided herein as Appendix 1. CDP-enabled devices share a number of information elements with their peers, via several CDP type-length-value (TLV) elements. In particular, a Capabilities TLV specifies a device type (such as router, switch, bridge, IP phone, etc), and can be used to determine a port role in the approaches herein. The Platform TLV provides a very specific device type, e.g., “Cisco IP Phone Model 7960” and therefore can be used to determine a role. Further, the sysName TLV specifies a system name, and can be used to determine a device identity herein. The Capabilities TLV, Platform TLV, and sysName TLV are examples of information that identifies or describes a network element and that can be used in approaches that use CDP. In various other embodiments, different information from CDP, or any other descriptive or

identification information that may be available from other protocols, may be used as a basis for determining a port role.

[0038] Devices that support CDP include a CDP database for locally storing information describing all immediately adjacent Layer 2 devices; software in the device can query the CDP database to determine the identity of neighbors. CDP is a one hop multi-cast protocol. Therefore, the CDP database in a device may contain information about more than one neighbor for a given port. For example, if a port on a first network element is attached to a Ethernet LAN hub network element, then all of the devices attached to that hub are recorded as neighbors of that port in the CDP database of the first network element. In this case, the profile for this port may not be as deterministic than if it happens to connect to a more intelligent second device.

[0039] In other embodiments, Simple Network Management Protocol (SNMP), Network-Based Application Recognition (NBAR), and Netflow protocols may be used to assemble information that can be used to determine port role or device identity. For example, if the first network element 102 has Internet Protocol (IP) connectivity to the second network element 112A, and the first network element has the SNMP read credentials for the second network element, then SNMP can be used to discover information identifying the second network element. For example, MIB II objects can be used to identify the second network element. In one approach, the sysDescr object may be used to derive an identity value, and the sysObjectID object may be used to determine a role.

[0040] In another approach, the protocols 802.1x, point-to-point protocol (PPP) and Cisco Service Selection Gateway (SSG) establish a user identity at the port level or circuit level, and then use that identity together with authorization information to allow IP connectivity or to select a specific type of connectivity service. For example, SSG provides a

user identifier of a logged-in user, group, or role as determined by an authentication, access and accounting (AAA) server, such as a RADIUS sever. In the absence of CDP information, the user identity information developed using these protocols may be used as an identity value herein, or the user identity information could augment the identity information that is available through CDP. However, these protocols provide user identity information rather than information identifying network elements. In addition, information accessible using from these protocols may reside at a network element that is several hops away from a first network element of interest.

[0041] The NBAR protocol extracts, from packets arriving on a port, information identifying an application that sent the packet. NBAR is a component of Cisco IOS®, and is generally described in the 1999 white paper, “Using Content Networking to Provide Quality of Service,” which is published at the Cisco public website, cisco.com. Information provided by NBAR may be used to determine the purpose of a packet. NBAR can detect packets from applications such as SAP, web browsers that generate HTTP traffic, telnet, email traffic that uses POP3, FTP, as well as routing protocols such as BGP, EIGRP, RIP. Information detected by NBAR is stored in a local NBAR database on a network device. Information in the NBAR database can be queried using “show” commands, an internal API, or through an SNMP MIB, or any other mechanism suitable for use by a management application.

[0042] Thus, NBAR is preconfigured to understand many or all of the protocol types that may be used to determine the role of a port. Using the information stored in the NBAR MIB, the role of a device attached to the port may be deduced. For example, if NBAR indicates that a particular port is carrying a high volume of HTTP traffic, then role agent 108 may determine that the port is connected to a Web server and apply an appropriate profile, role

and configuration. In contrast, if the port is carrying routing protocol packets, then role agent 108 can determine that the port is connected to a routing device.

[0043] NBAR does not collect source IP address information related to the application types so NBAR by itself is not deterministic. Thus, in one approach using NBAR, role agent 108 may generate a proposed role and provide a suggestion to a network administrator through a trap or other notification mechanism. Alternatively, the proposed role may be communicated to management application 120, for confirmation by the administrator.

[0044] In still another approach, the Netflow feature of Cisco IOS® is used as a basis for determining a role of a port. Netflow collects information about packets flowing through a device, including source IP address, source TCP/UDP port, and input interface. In one embodiment, discovery agent 106 scans the collected Netflow information and finds records for a given input interface that specify the type of traffic that is coming from the second network element that is connected to that interface. Based on such records, role agent 108 determines the role of the second network element. For example, the role agent 108 determines which well-known TCP/UDP port value is represented in the Netflow records, and then performs a table lookup in a mapping that associates port values to roles.

[0045] While CDP efficiently provides sufficient information for determining a role and identity of the second network element 112A, the alternate protocols that can be used to determine role and identity of a non-CDP second network element 112C may negatively impact performance of the first network element 102. Accordingly, in one approach, a default role is applied to a port or interface that does not have a CDP-enabled device attached.

[0046] Discovery agent 108 is programmable so that management application 120 or another external system can instruct the discovery agent what domain of objects on which to operate. For example, management application 120 may determine that it needs to determine

a role only for port 110B. In that case, management application 120 may send programming information or configuration information to discovery agent 108, or invoke a specified method or function of the discovery agent that is exposed to the management application through an API, to instruct the discovery agent only to process a particular port. The instructions and method calls also may instruct discovery agent 108 about what interface types to process, such as only Ethernet interfaces, only Fast Ethernet interfaces, serial interfaces, optical interfaces, etc., or what range of interfaces to process, e.g. only odd-numbered interfaces. The instructions issued by management application 120 may be based on network topology or other pertinent data.

[0047] Further, management application 120 may use the API or method invocations to deliver a set of rules that map Role and Identity information into a specific Profile. For example, a rule may specify:

IF type=any-router AND identity MATCHES 'london*' THEN profile=London-Profile

[0048] Referring again to FIG. 2, in block 203, the port of the first network element is associated with a port profile. In one embodiment, the result of processing by discovery agent 108 is information for each port that associates the port with a particular profile. Thus, the resulting information essentially indicates, for a port "xyz," that "port xyz *matches* profile abc", where "abc" is one of a plurality of stored profiles.

[0049] In block 204, the profile is associated with a port role definition. Role agent 108 performs block 204, in one embodiment. For example, role agent 108 maps profiles to roles that translate into configuration settings on the specified port. In block 206, one or more configuration values that are associated with the port role definition are received. Simple roles may result in retrieval of an enumeration of attribute-value pairs that need to be applied to a port. Attribute-value pairs may specify how to configure Voice VLAN, CAR, non-

forwarding of BPDU, protected port, ACLs, IP address, DHCP indications like scope for obtaining an IP address, etc.

[0050] In performing block 204 and block 206, role agent 108 may collect all the roles associated to the matched profile(s) or the port, and using fuzzy defaulting and precedence rules, create a collection of configuration attributes to be applied to or configured on the port. This procedure is now described in further detail with respect to FIG. 3A and FIG. 3B. FIG. 3A is a flow diagram that illustrates another embodiment of a method for role-based configuration of a port, and FIG. 3B is a flow diagram showing further steps in the method of FIG. 3A.

[0051] Referring first to FIG. 3A, in block 302, a first network device discovers a second device on a particular port of the first network device. Any of several discovery mechanisms can be used. For example, if the network elements support CDP, then the first network element 102 queries its CDP database for all entries associated with port 110A. If SNMP is used, then an SNMP query is issued to the second network element. If NBAR is used, then an SNMP query is issued to the NBAR MIB on the first network element.

[0052] In block 306, device-related Role and Identity information is extracted from the results of the operation performed in block 302. In the case of CDP, first network element 102 extracts the Capabilities TLV or Platform TLV and sysName TLV from the CDP database for port 110A. The Capabilities TLV or Platform TLV is used to determine a device type, and the sysName TLV specifies a system name that is used to determine a device identity. Blocks 302, 304, and 306 may be performed by discovery agent 106.

[0053] In block 307, a profile is selected from among profile rules 130 based on the discovered information. In block 308, a profile rule is applied to the device information. By use of the approach herein in which a port is used to match a profile and a profile is

associated with a role, enables the system to self-configure ports for many different purposes.

As an example, assume that profile rules 130 comprise the following rules:

1. “London-profile” → other-end-hostname MATCHES ‘london.*’ AND other-end-devtype == any-router
2. “Dubai-profile” → other-end-hostname MATCHES ‘dubai.*’ AND other-end-devtype == any-router
3. “Core-Ring-profile” → other-end-devtype == any-router AND wire-type == GigaEthernet AND NOT (other-end-hostname MATCHES ‘london.*’) AND NOT (other-end-hostname MATCHES ‘dubai.*’)

[0054] While in this example each profile maps to a different role, in other examples, many profiles can map to the same role. Assume further that role mappings 140 comprises:

1. Link-to-London.role → ip-addr = unnumbered; IP-unnum-param = &Core-Uplink-role.interface.7
2. Link-to-Dubai.role → ip-addr = 144.23.34.52; netw-mask = /30; ACL= 222
3. Core-Uplink-role => → interface-type = dot1Q; subinterface=(VLAN = 7; ip-addr = 144.23.4.42; netw-mask = /30); subinterface = (VLAN =8, ip-addr = 144.23.34.53; netw-mask = /30; ACL= 222)

[0055] Using these definitions, after discovery is performed at block 202, the discovered profiles of ports are matched to roles maintained by the role agent 108, and specific sets of changes to port configurations are made automatically. Thus, in this example block 307 and block 308 would involve applying each of the rules defined above to the discovered sysName (Identity) and Capability or Platform (Role) values.

[0056] An approach that relies on use of the CDP sysName TLV for the identity of a connected device assumes that values for the CDP sysName TLV are set by network

administrators and are generally unique within the domain of their control. Such network administrators are expected to create profile rules 130 and to ensure that each profile is unique with respect to identity information. If multiple devices are given the same sysName value and thus have the same identity information, then such devices potentially could have the same profiles applied.

[0057] Applying the same profiles could be a desired behavior, or it could be an accident. For example, an administrator may wish to have multiple identities so that one profile is applied to more than one port. In one embodiment, the system of FIG. 1 may include a repository of information about which profiles have been applied to which ports. Further, the system of FIG. 1 and the processes of FIG. 2, FIG. 3A-3B may maintain a counter reflecting a use count on each profile such that a profile may be used once and only once. In one embodiment, the counter tracks the number of ports to which a profile is currently mapped. Thus, when an IP phone is plugged into a port, the counter for the phone profile is incremented, and when the IP phone is unplugged the counter decrements; if two IP phones are currently plugged in then the value of the counter is 2. This has implications for memory use on devices with many ports or profiles.

[0058] To allow a single profile to be assigned to multiple identities, discovery agent 106 can process wild cards in its matching logic. For example, assume that an organization has a policy that all corporate printers are given a name of the form "<building-location>-printer" and that the corporate printers are all CDP enabled. Discovery agent 106 is given a profile of the form:

Corporate-Printer → Type=printer and Identity="*-printer"

[0059] Using CDP information collected by the switch to which the printers are connected, any device information that matched the foregoing rule is assigned a Corporate-

Printer profile. As a result, the Corporate-Printer profile is used for as many printers as match the profile rule.

[0060] As shown in block 310, a test is performed to determine whether the discovered information satisfies a particular rule. If not, then in block 312 a test is performed to determine whether other profiles are available for testing. If the test of block 312 is true, then control returns to block 308 and block 310 for processing of the other profiles. If the test of block 312 is negative, then in block 316 the then-current port is mapped to a default profile.

[0061] If a match occurs at block 310, then control transfers to block 314, in which the then-current port is mapped to a profile selected from among profile rules 130.

[0062] Referring now to FIG. 3B, in block 318, the role associated with the mapped profile is retrieved. For example, role agent 108 retrieves a role from among stored role mappings 140. In block 320, configuration values associated with the role are retrieved from storage. In block 322, the retrieved configuration values are applied to the then-current port.

[0063] Using this approach, a port is automatically configured according to a role played in the network by a device connected to the port, and the automatic configuration may be dynamic. For example, if a port is connected to a router and that router identifies itself as London (the London profile), then a set of configurations (the “London role”) are applied to the port. If at some later time, the same port is connected to a router identified with Dubai, then the Dubai role is applied to the port. As a result of using the approaches herein, in which a network element can make configuration changes to itself automatically based on information collected about its neighbor devices, the cost of network management due to accidental changes to connections to a port is eliminated or greatly reduced, because the network device adapts to the change automatically.

[0064] As another example, assume that network element 102 is a local area network switch that has multiple physical ports attached to several different types of devices such as printers, servers, uplink to corporate backbone, desktop computers, IP phones, etc. Based on the type of the device on the other end of a port, the port configures itself for specific QoS policies, a VLAN assignment, etc. If two network cables are physically switched, e.g., if a port connected to a server and a port connected to a desktop are switched, then the ports are reconfigured automatically to configurations that are specific to the device to which the ports now connected. No network engineer is needed to reconfigure the ports to match the change in physical connectivity or to physically change the connections back to the original configuration.

2.3 COMPLETE EXAMPLE

[0065] A complete example is now provided with reference to FIG. 4 and FIG. 5. FIG. 4 is a block diagram of an example device port profile. FIG. 5 is a block diagram of an example device port role definition. In this example, profiles and roles for building switch port settings are described.

[0066] Assume that a two-story office building is part of a corporate complex and is comprised of offices, cubicles, and conference rooms. Each floor has four network switches and the building has one router that interconnects the switches to a corporate network backbone. Each switch has 24 Ethernet/Fast Ethernet/Gigabit Ethernet ports, of which 23 ports are for local devices and one port is for uplink to the backbone router. Assume further that all Cisco switches and routers are used, all printers are HP CDP-capable printers, and the rooms include IBM PCs and Cisco IP telephones.

[0067] Further assume that all devices in the building follow standard naming conventions. For example, PCs are all named <user-id><#>-PC. All printers are named

<building-floor-location>-PRINTER. All IP-phones in the building are named <user-id><#>-PHONE. All switches are named <building-floor-location>-SWITCH. All backbone routers are named <building-location>-BBROUTER. All the switches, routers, IP phones and printers are CDP enabled. The PCs are not CDP enabled.

[0068] Assume that the default switch port configuration is:

Bandwidth=10mb

QOS=<pc qos settings>

Also assume that the profiles shown in FIG. 4 and the roles shown in FIG. 5 are programmed into the switches.

[0069] Now assume that Switch 1 of Building 1 boots and initiates the approaches herein. Switch 1 supports CDP and has a port designated Port 1 that is connected to an IP phone that an administrator previously named “astamler7-PHONE”. The IP phone, which is CDP-enabled, periodically issues a CDP announcement packet that includes a Capabilities TLV device type value indicating “VOIPphone” and a sysName TLV value indicating “astamler7-PHONE”. Switch 1 updates its CDP database upon receiving the CDP announcement. Discovery agent 106 queries the CDP database for information associated with Port 1, applies the profile rules of FIG. 4 and finds a match for a profile name 402 of PHONE 402B, because the rule “TYPE=VOIPphone” 404B is TRUE for the received Capabilities TLV value. Therefore, discovery agent 106 determines that Port 1 has role 406B, the “PHONE-ROLE.”

[0070] Role agent 108 of Switch 1 then retrieves role mappings 140 of FIG. 5. Role agent 108 determines that PHONE-ROLE 502B of FIG. 5 is associated with configuration settings 504B. Therefore, role agent 108 applies the indicated configuration settings to Port 1. As a result, Port 1 is set with a bandwidth of 100mb, and specified QoS values.

[0071] Switch 1 then proceeds to query the CDP database, determine a profile and role, and apply a configuration to ports 2-23. Assume that a query to the CDP database for Port 2 results in receiving a sysName TLV having a value of “IDENTITY=JUPITER-SERVER.” The profile rules 130 of FIG. 4 have no entry for such an identity value. Therefore, discovery agent 106 fails to find a match against any stored profile. As a result, discovery agent 106 assigns a default role to Port 2. Further, role agent 108 maps the default role to the default configuration, and applies the default configuration to Port 2.

[0072] The foregoing process is repeated for all other ports of Switch 1. Other switches perform a similar process. The process may be repeated if ports become inactive and re-activate, or according to a specified schedule. Using these profiles and roles, users may randomly change connections to ports of the building switches, and the ports always will have the right configurations based on evaluation of the profiles and rules using the approaches described above.

2.4 TIME OF EXECUTION

[0073] The discovery agent 106 may execute at different times and for different ranges of ports of a first network element. For example, in one embodiment the discovery agent 106 executes at device boot time for all targeted ports of a device. In another embodiment, the discovery agent 106 executes only when a port initiates operation, and only configures the port that just initiated operation. In yet another embodiment, the discovery agent 106 executes according to one or more specified time periods, and configures all targeted ports. In general, the role of a port is expected to change relatively infrequently, so that discovery agent 106 most often will execute in response to boot and status change events. Ports that do change often could be excluded from a list of ports to evaluate in a particular network element. For example, ports of a device terminating at a frequently used conference room,

such that many different PCs and other devices are regularly connected to the ports, could be excluded from the list of ports to evaluate.

2.5 INTERACTION WITH MANAGEMENT APPLICATIONS AND OTHER SOLUTIONS

[0074] Optionally, the discovered port information is provided to management application 120. The management application 120 can use the discovered port information to aid in the creation of topology-specific configuration files. Alternatively, management application 120 can instruct the role agent 108 to apply configuration changes to ports as they are discovered.

[0075] The approaches herein may be integrated with other network management solutions and systems. For example, CNS can deliver the roles and profiles to the device as part of a configuration file, or can process information returned by the discovery agent 106 to make configuration changes, and send such changes back to the device 102. Further, device identity information that is developed by the Cisco Service Selection Gateway (SSG) in its normal operation may be used to provide or augment the identity information used by the approaches herein.

2.6 APPLICABILITY TO PHYSICAL PORTS AND LOGICAL PORTS

[0076] The approaches herein are applicable to physical ports and logical ports. In one embodiment, applying the approaches to physical ports is expected to yield the most value for the least CPU utilization. In this approach, CDP can be used to find neighbor network elements of a port, and type and identity information for the neighbor elements. In the case of a one-to-many device CDP neighbor mapping for a port, the profile of that port becomes less deterministic. If CDP is not available, then SNMP, NBAR or Netflow may be used. Use of NBAR or Netflow involves packet sniffing, and use of SNMP requires the first network

element to have proper SNMP credentials to query neighbor devices. Further, with SNMP, the identity of neighbor devices could be determined by packet sniffing or by providing device information to the discovery agent 106.

[0077] When the approaches herein are used with a logical port, packet sniffing is performed to determine the types of logical traffic on a port. For example, packet sniffing might indicate that traffic over a port is marked with VLAN identifiers, ATM permanent virtual circuits (PVCs), etc. If either VLAN or PVC traffic is detected, then the port is associated with a profile related to VLANs or PVCs.

2.7 FAILURE PROCESSING

[0078] In one embodiment, role agent 108 includes logic for processing profile mismatches. Alternatively, error detection is a manual process. In this approach, if incorrect information is received through CDP or another protocol, then an incorrect role may be selected and incorrect configuration changes may be attempted on a given port. Once failures are detected, rule sets can be modified or removed so that the misconfiguration does not happen again in a particular case.

3.0 IMPLEMENTATION MECHANISMS – HARDWARE OVERVIEW

[0079] FIG. 6 is a block diagram that illustrates a computer system 600 upon which an embodiment of the invention may be implemented. The preferred embodiment is implemented using one or more computer programs running on a network element such as a router device. Thus, in this embodiment, the computer system 600 is a router.

[0080] Computer system 600 includes a bus 602 or other communication mechanism for communicating information, and a processor 604 coupled with bus 602 for processing information. Computer system 600 also includes a main memory 606, such as a random access memory (RAM), flash memory, or other dynamic storage device, coupled to bus 602

for storing information and instructions to be executed by processor 604. Main memory 606 also may be used for storing temporary variables or other intermediate information during execution of instructions to be executed by processor 604. Computer system 600 further includes a read only memory (ROM) 608 or other static storage device coupled to bus 602 for storing static information and instructions for processor 604. A storage device 610, such as a magnetic disk, flash memory or optical disk, is provided and coupled to bus 602 for storing information and instructions.

[0081] A communication interface 618 may be coupled to bus 602 for communicating information and command selections to processor 604. Interface 618 is a conventional serial interface such as an RS-232 or RS-422 interface. An external terminal 612 or other computer system connects to the computer system 600 and provides commands to it using the interface 614. Firmware or software running in the computer system 600 provides a terminal interface or character-based command interface so that external commands can be given to the computer system.

[0082] A switching system 616 is coupled to bus 602 and has an input interface 614 and an output interface 619 to one or more external network elements. The external network elements may include a local network 622 coupled to one or more hosts 624, or a global network such as Internet 628 having one or more servers 630. The switching system 616 switches information traffic arriving on input interface 614 to output interface 619 according to pre-determined protocols and conventions that are well known. For example, switching system 616, in cooperation with processor 604, can determine a destination of a packet of data arriving on input interface 614 and send it to the correct destination using output interface 619. The destinations may include host 624, server 630, other end stations, or other routing and switching devices in local network 622 or Internet 628.

[0083] The invention is related to the use of computer system 600 for providing role-based configuration of a port of a network element. According to one embodiment of the invention, role-based configuration of a port of a network element is provided by computer system 600 in response to processor 604 executing one or more sequences of one or more instructions contained in main memory 606. Such instructions may be read into main memory 606 from another computer-readable medium, such as storage device 610. Execution of the sequences of instructions contained in main memory 606 causes processor 604 to perform the process steps described herein. One or more processors in a multi-processing arrangement may also be employed to execute the sequences of instructions contained in main memory 606. In alternative embodiments, hard-wired circuitry may be used in place of or in combination with software instructions to implement the invention. Thus, embodiments of the invention are not limited to any specific combination of hardware circuitry and software.

[0084] The term “computer-readable medium” as used herein refers to any medium that participates in providing instructions to processor 604 for execution. Such a medium may take many forms, including but not limited to, non-volatile media, volatile media, and transmission media. Non-volatile media includes, for example, optical or magnetic disks, such as storage device 610. Volatile media includes dynamic memory, such as main memory 606. Transmission media includes coaxial cables, copper wire and fiber optics, including the wires that comprise bus 602. Transmission media can also take the form of acoustic or light waves, such as those generated during radio wave and infrared data communications.

[0085] Common forms of computer-readable media include, for example, a floppy disk, a flexible disk, hard disk, magnetic tape, or any other magnetic medium, a CD-ROM, any other optical medium, punch cards, paper tape, any other physical medium with patterns of holes, a

RAM, a PROM, and EPROM, a FLASH-EPROM, any other memory chip or cartridge, a carrier wave as described hereinafter, or any other medium from which a computer can read.

[0086] Various forms of computer readable media may be involved in carrying one or more sequences of one or more instructions to processor 604 for execution. For example, the instructions may initially be carried on a magnetic disk of a remote computer. The remote computer can load the instructions into its dynamic memory and send the instructions over a telephone line using a modem. A modem local to computer system 600 can receive the data on the telephone line and use an infrared transmitter to convert the data to an infrared signal. An infrared detector coupled to bus 602 can receive the data carried in the infrared signal and place the data on bus 602. Bus 602 carries the data to main memory 606, from which processor 604 retrieves and executes the instructions. The instructions received by main memory 606 may optionally be stored on storage device 610 either before or after execution by processor 604.

[0087] Communication interface 618 also provides a two-way data communication coupling to a network link 620 that is connected to a local network 622. For example, communication interface 618 may be an integrated services digital network (ISDN) card or a modem to provide a data communication connection to a corresponding type of telephone line. As another example, communication interface 618 may be a local area network (LAN) card to provide a data communication connection to a compatible LAN. Wireless links may also be implemented. In any such implementation, communication interface 618 sends and receives electrical, electromagnetic or optical signals that carry digital data streams representing various types of information.

[0088] Network link 620 typically provides data communication through one or more networks to other data devices. For example, network link 620 may provide a connection

through local network 622 to a host computer 624 or to data equipment operated by an Internet Service Provider (ISP) 626. ISP 626 in turn provides data communication services through the worldwide packet data communication network now commonly referred to as the "Internet" 628. Local network 622 and Internet 628 both use electrical, electromagnetic or optical signals that carry digital data streams. The signals through the various networks and the signals on network link 620 and through communication interface 618, which carry the digital data to and from computer system 600, are exemplary forms of carrier waves transporting the information.

[0089] Computer system 600 can send messages and receive data, including program code, through the network(s), network link 620 and communication interface 618. In the Internet example, a server 630 might transmit a requested code for an application program through Internet 628, ISP 626, local network 622 and communication interface 618. In accordance with the invention, one such downloaded application provides for role-based configuration of a port of a network element as described herein.

[0090] The received code may be executed by processor 604 as it is received, and/or stored in storage device 610, or other non-volatile storage for later execution. In this manner, computer system 600 may obtain application code in the form of a carrier wave.

4.0 EXTENSIONS AND ALTERNATIVES

[0091] In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

APPENDIX 1—CISCO DISCOVERY PROTOCOL

Objectives

- cisco devices (and other devices which implement CDP) need to discover each other in a protocol/media independent way.
- Network management applications need to dynamically discover cisco devices which are neighbors of already-known devices, especially neighbors which implement lower-layer "transparent" protocols.
- Network management applications can retrieve the device-type and SNMP-agent address of neighboring devices. This enables the applications to subsequently send SNMP queries to the neighboring devices.
- There are a number of other protocols which need to know if and when that protocol is enabled on a neighboring device. Rather than have each such protocol send its own periodic "hello" message, such "hello" messages can be piggy-backed on CDP packets.

Introduction

CDP is a device discovery protocol that runs on all cisco manufactured equipment (i.e. routers, bridges, communication servers, WBU switches). Each device sends periodic messages to a multicast address. Each device listens to the periodic messages sent by others in order to learn about neighboring devices and determine when their interfaces to the media go up or down. A second version of the protocol, CDPv2, uses a new value in the version field, introduces new TLVs, and tightens the rules for handling additional definitions.

Data points

CDP will be assigned a cisco HDLC protocol type value. There will be a cisco proprietary SNAP value that enumerates HDLC protocol type values so CDP is able to run on all media that support SNAP (i.e. LAN media, Frame Relay, ATM).

CDP does not run on top of any network layer. It runs over the the data link only. Therefore two systems that support different network layer protocols can learn about each other.

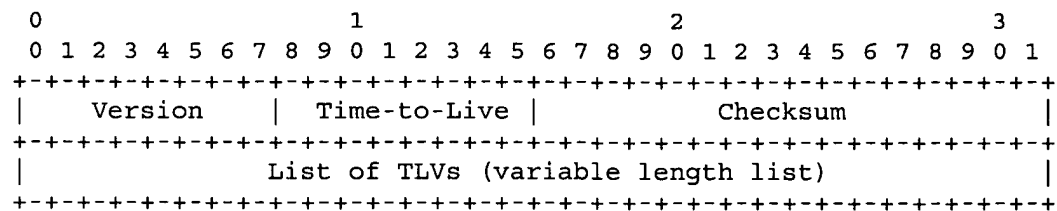
HDLC protocol type for CDP is 0x2000.

CDP will send packets on LANs with the multicast address 0100.0ccc.cccc, except on 802.5 networks where the packets are sent to the address c000.0800.0000. The SNAP format is:

LLC	Org ID	HDLC protocol type
0xaaaa03	0x00000c	0x2000

A packet addressed to this CDP multicast address but with a different 8-byte SNAP value, should be processed according to that SNAP value, or discarded if the SNAP value is unknown. If CDP is disabled, packets received at the CDP address with CDP's SNAP value must be discarded (not forwarded).

Packet Format

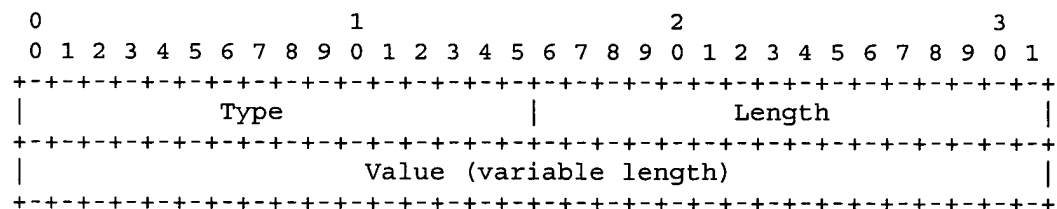


Version (8 bits): CDPv1 - 0x01
CDPv2 - 0x02

Time-to-live (8-bits): The length of time in seconds a receiver should keep the information in this packet.

Checksum (16-bits): The 1's complement of the 1's complement sum. The standard IP checksum (with the modification that the odd byte at the end of an odd length message is used as the signed low 8 bits of an extra word, rather than as the unsigned high 8 bits.)

TLV format:



Type (16-bits): Well known values defined in this document. If an implementation does not understand a Type value it skips over it using the length field.

Length (16-bits): Length in bytes of the Type, Length, and Value fields.

Device-ID TLV: 0x0001

Identifies the device. This is used so different address references can be associated with the same device. This is in the form of a character string. The TLV length determines the length of the string. Previous versions of this specification suggested that this TLV be the sending device's FQDN (Fully Qualified Domain Name, e.g., dent.cisco.com); however, practical experience has shown that devices do not always have unique FQDNs in some environments. Therefore, a new TLV (the sysName-TLV, see below) has been defined for the FQDN, and the recommended value for this Device-ID TLV has changed to be (an ASCII string containing) the hardware serial number or a MAC address, which is unique to the sending device.

When the Device-ID contains the MAC address ID, the canonical address must be encoded as an ASCII string of 12 hexadecimal digits with no separators embedded within, e.g. "00d0061d200a" for a MAC address whose OUI is 00:d0:06. The letters a-f may be in upper or lower case.

When the Device-ID contains a hardware serial number, it should be encoded as an ASCII string consisting of the serial number, followed by a '/' separator, followed the manufacturer's OUI in canonical form as an ASCII string. The manufacturer's OUI, if it contains any hex digits A-F, must be encoded in upper case. All characters leading up to the first '/' will be treated as is, and should constitute a globally unique string when combined with the manufacturer's OUI. Since the manufacturer's serial number may be case-sensitive, any comparisons between two such Device-ID strings should be case sensitive. The manufacturer's serial number must not contain an embedded '/'. E.g. a Cisco product with the chassis serial number "SCA0307027E" would be encoded as "SCA0307027E/00000C".

Address TLV: 0x0002

Contains a list of network layer addresses encoded in the same manner as used by IDRP [1]. If the device is SNMP-manageable, the first address in the list must be an address at which the device will receive SNMP messages. If the device can receive SNMP messages the list must be non-empty. The addresses advertised are typically ones assigned to the interface the CDP packet is sent on. A device may advertise all of its addresses but is not required to. A device may optionally advertise one or more Loopback IP addresses.

Format :

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
+										+										+										+									
										Number of addresses																													
+										+										+										+									
										IDRP encoded address																													
+										+										+										+									

An address is encoded in IDRP format:

[illegible]

```
PT: Protocol type
    1 = NLPID format
    2 = 802.2 format
```

164 PT Length:
165 Length of protocol field, 1 for PT = 1, and either 3 or 8
166 for 802.2 format depending if SNAP is used for PT = 2.
167
168 Protocol:
169 0x81 - CLNS (PT = 1)
170 0xCC - IP (PT = 1)
171 0xAAAA03 000000 0800 - IPv6 (PT = 2)
172 0xAAAA03 000000 6003 - DECNET Phase IV (PT = 2)
173 0xAAAA03 000000 809B - AppleTalk (PT = 2)
174 0xAAAA03 000000 8137 - Novell IPX (PT = 2)
175 0xAAAA03 000000 80c4 - Vines (PT = 2)
176 0xAAAA03 000000 0600 - Xerox XNS (PT = 2)
177 0xAAAA03 000000 8019 - Apollo Domain (PT = 2)
178
179 Address length:
180 Length of address field in bytes.
181
182 Address:
183 Address of interface or system if addresses not assigned to
184 interfaces. For CLNS, the address contains a 1-byte length,
185 in bytes (not in bits, as some interpretations of [1] might
186 indicate), followed by the address.
187
188 Port-ID TLV: 0x0003
189 Identifies the port the CDP packet is sent on. This is encoded
190 as an ASCII character string. The TLV length determines the
191 length of the string. Existing implementations should, and all
192 new implementations must, have the value of this TLV be the same
193 as the MIB object ifName for the ifTable entry on which the CDP
194 packet is sent (e.g., "Ethernet0").
195
196 Capabilities TLV: 0x0004
197 Describes the device's functional capability.
198
199 Format:
200 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
201 +-----+-----+-----+-----+-----+-----+-----+-----+
202 |
203 |
204 +-----+-----+-----+-----+-----+-----+-----+-----+
205
206 Bit definitions:
207
208 Router: 0x01
209 Currently performing level-3 routing for at least one
210 network layer protocol.
211
212 TB Bridge: 0x02
213 Currently performing level-2 transparent bridging.
214
215 SR Bridge: 0x04
216 Currently performing level-2 source route bridging. An SRT
217 bridge would set both this bit and the TB Bridge bit.
218

219 Switch: 0x08
 220 Provides layer-2 and/or layer-3 switching.
 221
 222 Host: 0x10
 223 Sends and receives packets for at least one network layer
 224 protocol. If the device is routing the protocol, this
 225 bit should not be set.
 226
 227 IGMP conditional filtering: 0x20
 228 The Bridge or Switch does not forward IGMP Report packets on
 229 non router-ports.
 230
 231 Repeater: 0x40
 232 Provides level-1 functionality.
 233
 234 VoIP Phone: 0x80
 235 Provides voice or data telephony capability. A device
 236 which sets this bit should handle the VoIP phone value
 237 of the Appliance VLAN-ID TLV on a received CDP packet.
 238
 239 Version TLV: 0x0005
 240 Contains information about the software release version the
 241 device is running. This is in the form of a character string.
 242 The TLV length determines the length of the string. This is
 243 the information returned in "show version".
 244
 245 Platform TLV: 0x0006
 246 Describes the hardware platform of the device. Encoded as an
 247 ASCII character string. The TLV length determines the length
 248 of the string. Recommended string values include:
 249
 250 cisco RSP cisco 2000 cs500
 251 cisco RP1 cisco 1000 WS-C5000
 252 cisco 7010 cisco AGS+ LS100
 253 cisco 4500 cisco CSC3 WS-C6009
 254 cisco 4000 cisco CSC4 cisco 2505
 255 cisco 3100 cisco MGS etc.
 256 cisco 3000 cisco CGS
 257 cisco 2500 cisco IGS
 258
 259 IP Network Prefix TLV: 0x0007
 260 This TLV is used by On Demand Routing (ODR). When transmitted
 261 by a hub router, this TLV is 4 bytes long and contains a
 262 default route (an IP address). When transmitted by a stub
 263 router, this TLV contains a list of network prefixes of stub
 264 networks to which the sending stub router can forward IP
 265 packets. Each network prefix is formatted as a 4-byte IP
 266 address followed by a 1-byte net mask length. Thus, the
 267 length of the value sent by a stub router is a multiple of 5
 268 bytes.
 269
 270 Protocol-Hello TLV: 0x0008
 271 Specifies that a particular protocol has asked CDP to piggyback
 272 its "hello" messages within transmitted CDP packets. The
 273 value of this TLV protocol has a length greater or equal to 5

and less than or equal to 32 bytes. The first 5 bytes are the protocol's 5-byte SNAP value, contains three bytes of OUI followed by two bytes of protocol-id. Depending on the protocol identified, additional bytes may be appended as part of the value. (Note that the protocol-id value is an EtherType value when the OUI is zero, and no additional bytes are appended.) Multiple Protocol-Hello TLVs, each for a different protocol, may be included in one CDP packet.

VTP Management Domain TLV: 0x0009

For a device running the VTP protocol, contains the name of the VTP management domain (on the particular interface on which the CDP packet is sent). The TLV length determines the length of the VTP management domain name. A length of zero indicates that a device is running VTP but has no management domain name assigned.

Native VLAN TLV: 0x000a

Some VLAN-tagging mechanisms, such as 802.1Q, support links on which some frames are tagged to indicate which VLAN the frame belongs to, and some frames are untagged. Such untagged frames are said to be on the "native" VLAN of the link. This TLV, if included, indicates the ISL number of the native VLAN of the interface on which the CDP packet is sent, whether or not that VLAN is enabled on the link, and whether the link is a trunk or a host/edge port.

Full/Half Duplex TLV: 0x000b

Point-to-point Ethernet interfaces can run in full-duplex mode, or in half-duplex mode. It is a common cause of network problems for the two ends of a link to be in different modes. This TLV allows devices to recognize this error situation. The TLV value is one byte in length with its least significant bit indicating the mode: a '1' indicates full-duplex, '0' indicates half-duplex. All other bits of the TLV value should be set to '0' on transmission and ignored on receipt. This TLV is expected to be used on Ethernet interfaces; its use on other types of interfaces is optional.

(Note that TLVs: 0x000c and 0x000d are not used.)

Appliance VLAN-ID TLV: 0x000e

A number of tuples, each containing an Appliance ID (1 byte) and a VLAN-id (2bytes).

Appliance id:

0x00 Reserved
0x01 VoIP phone
0x02 - 0xff = for future use.

VLAN-id (for VoIP phone):

When a local 802.1Q interface has been configured to send and receive VoIP (and related) packets on a particular VLAN, this TLV contains the setting of the local configuration. Devices receiving this TLV may, if appropriate, adjust their

```

329 configuration to match this setting. The value has the
330 following meanings:
331
332 0 - VoIP and related packets are expected to be sent
333 and received with VLAN-id=0 and an 802.1p priority
334 N - VoIP and related packets are expected to be sent
335 and received with VLAN-id=N and an 802.1p priority,
336 for N=1,...,4094
337 4095 - VoIP and related packets are expected to be sent
338 and received untagged, without an 802.1p priority.
339
340 Trigger TLV: 0x000f
341 There are times when a device receiving a CDP packet does more
342 with the information that just store it in its MIB. For such
343 uses, it is often advantageous for the receiving device to
344 receive a CDP packet immediately rather than waiting for the
345 sender's next transmission based on the periodic transmission
346 interval. For such a situation, the receiving device can
347 include a Trigger TLV in a CDP packet it sends.
348
349 A CDP packet containing a Trigger TLV is called a "trigger".
350 Under certain circumstances, a device receiving a trigger
351 will generate and transmit (on the same interface) its own
352 CDP packet as a response. The Trigger TLV has several
353 options, defined below, which specify the circumstances under
354 which a response is generated. A device must take care not
355 to include the Trigger TLV in all periodic CDP packet it
356 sends, i.e., triggers must not be sent periodically.
357
358 A device must delay for a random amount of time before sending
359 a trigger (for point-to-point links, see below). If during
360 that delay, it receives the same kind of trigger sent by some
361 other device then it may cancel its delayed trigger, assuming
362 it will get a response due to the received trigger. If not,
363 and during the delay, the device receives a CDP packet which
364 contains the information which it was intending to get by
365 sending the trigger, then the delayed trigger may be
366 cancelled/not sent. If the delayed trigger is not cancelled,
367 then it is sent at the end of the delay period.
368
369 There are two formats for the value of a Trigger TLV. The first
370 format is:
371
372 0 1 2 3
373 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
374 +-----+-----+-----+-----+-----+-----+-----+-----+
375 | Type | Device-id length | Device-id field |
376 +-----+-----+-----+-----+-----+-----+-----+-----+
377 | from which response requested |
378 +-----+-----+-----+-----+-----+-----+-----+-----+
379
380 Type = '0001'b
381 Device-id length = the length of the immediately following
382 Device-id field.
383 Device-id field = a device only responds to this type of trigger,

```

384 if the value of the device's Device-id TLV
 385 exactly matches the value of this field.
 386

387 The second format is:

```

388
389      0                1                2                3
390      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
391      +-----+-----+-----+-----+-----+-----+-----+-----+
392      | Type   | Info-list length | Info-list field |
393      +-----+-----+-----+-----+-----+-----+-----+-----+
394      |
395      +-----+-----+-----+-----+-----+-----+-----+-----+
396

```

397 Type = '0002'b
 398 Info-list length = the length of the immediately following
 399 Info-list field; a length of 0 is allowed.
 400 Info-list field = the concatenation of zero or more 2-byte
 401 sub-fields, such that each 2-byte sub-field
 402 identifies a particular type of information.
 403 If this list is non-empty, then a device
 404 must not respond to the trigger unless the
 405 CDP packet it sends in response will
 406 contain all the indicated types of information.
 407 The meaning of the 2-byte fields are:

```

408
409      0x0000 = Reserved
410      0x0001 = a VLAN-id (for VoIP phone)
411      0x0002 - 0xffff = for future use.
412

```

413 When a device receives a trigger containing a Device-id field
 414 which exactly matches the device's Device-id value (in the
 415 last CDP packet the device sent), then the device immediately
 416 sends a CDP packet in response.

417
 418 When a device receives a trigger and the CDP packet it would
 419 send in response would contain all the information identified
 420 by the trigger's Info-list or if the length of the Info-list
 421 is zero, then it starts a delay timer. While this timer is
 422 running, if the device receives a CDP packet containing all
 423 the information identified by the trigger's Info-list, then
 424 the device may stop the timer, and forget it received the
 425 trigger. When the timer expires, the device sends a CDP
 426 packet.

427
 428 Since a response to a trigger containing an Info-List is only
 429 sent by a device which can supply all the requested information,
 430 it is advisable for an Info-List to contain the minimum
 431 amount of information required to be contained in a response.
 432 When a device receives a trigger containing multiple Trigger
 433 TLVs, it evaluates each Trigger TLV in turn, and generates a
 434 CDP packet in response if doing so will satisfy any one or
 435 more of the Trigger TLVs.

436
 437 Two delay timers are specified above: a delay timer before
 438 sending a trigger, and a delay timer before sending a

response to certain types of triggers. The timer value for both of these delay timers must be a random value in the interval (0..N) seconds. For point-to-point links, N is always zero (i.e., zero delay). For other media, N has the value 1 second.

Regardless of how many triggers are sent/received, a device must not send more than two CDP packets in any one second on any one interface.

Power Consumption TLV: 0x0010

It's possible for some devices (e.g., IP Phones) to obtain power via their network connectivity. In such situations, it's useful for the system supplying power through the network to know how much of that power is being consumed. The value contained in a Power Consumption TLV, specified either as a 16-bit or a 32-bit value, is the maximum amount of power, in milliwatts, expected to be obtained and consumed from the interface over which the CDP packet is sent.

MTU TLV: 0x0011

On types of sub-networks where the MTU size is configurable, it is possible for devices connecting to it to be configured with different MTU values. This TLV allows devices to recognize this error situation. The TLV value is four bytes in length and contains the MTU of the interface via which this TLV is transmitted. MTU is defined as the size of the largest datagram which can be sent/received on the interface, specified in octets. For interfaces that are used for transmitting IP datagrams, this is the size of the largest IP datagram that can be sent/received on the interface.

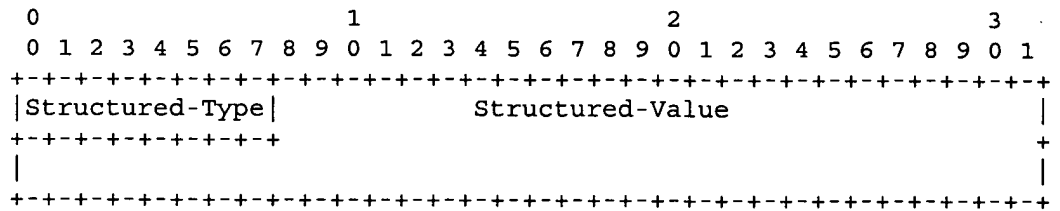
Extended Trust TLV: 0x0012

Some network-edge devices (e.g., in the Wiring Closet) provide the ability to classify traffic received on an interface, e.g., for the purpose of assigning packets to a QoS class and marking them accordingly. However, sometimes, a network administrator is willing to trust a Host/Server to mark the packets itself, in which case, the port to which that Host/Server is connected is said to be "trusted", and packets received on such a port are not re-marked. When one port of a simple switching device is plugged into a trusted port of a larger switch, it can be useful to extend the concept of trust from the larger switch port to the other (external) port(s) of the simple switching device without having to explicitly configure the simple switching device itself. (One example of such a simple switching device is an IP phone containing an internal 3-port switch.) The TLV value is one byte in length with its least significant bit indicating the trust mode: a '1' indicates extended trust, '0' indicates the absence of extended trust. All other bits of the TLV value should be set to '0' on transmission and ignored on receipt. A CDP packet without this TLV also indicates the absence of extended trust.

COS for Untrusted Ports TLV: 0x0013

494 This TLV only has any meaning in a CDP packet containing a
 495 Extended Trust TLV which indicates the absence of extended
 496 trust.
 497 The TLV value is one byte in length and contains a layer-2
 498 ClassOfService value, e.g., an 802.1D/802.1p priority value.
 499 This is the COS value with which all packets received on an
 500 untrusted port should be marked by a simple switching device
 501 which cannot itself classify individual packets.
 502
 503 sysName TLV: 0x0014
 504 An ASCII string containing the same value as the sending
 505 device's sysName MIB object. sysName was originally defined
 506 in MIB-II, and it's most recent definition is in RFC 1907. The
 507 definition specifies that its value is, by convention, the Fully
 508 Qualified Domain Name of the device (e.g. dent.cisco.com), or
 509 if the name is unknown, the value is the zero-length string.
 510
 511 sysObjectID TLV: 0x0015
 512 This TLV contains the OBJECT-IDENTIFIER value of the sending
 513 device's sysObjectID MIB object. sysObjectID was originally
 514 defined in MIB-I, and it's most recent definition is in RFC
 515 1907. The value of the TLV is the ASN.1/BER-encoding of the OID,
 516 i.e., the OID 1.3.6.1.4.1.9.1.64 is encoded as the hex string:
 517
 518 06 08 2b 06 01 04 01 09 01 40
 519
 520 Management-Address TLV: 0x0016
 521 Contains a list of network layer addresses encoded in the same
 522 manner as the Address-TLV (see above). When present, this TLV
 523 contains a list of all the addresses at which the device will
 524 accept SNMP messages, including those it will only accept when
 525 received on interface(s) other than the one over which the CDP
 526 packet is being sent. If the set of all such addresses is too
 527 large to be included in one CDP packet, then an address can be
 528 omitted providing that all interfaces, on which that address is
 529 being accepted, are accepting at least one of the remaining
 530 addresses. A device which is not (currently) manageable via
 531 any network protocol should so indicate by sending this TLV
 532 containing one and only one address: the special value of the
 533 IPv4 address 0.0.0.0. The first address in the list is
 534 considered to be the primary management address of the device
 535 itself, which is independent of the interface on which the
 536 CDP message is sent, and therefore should be the same in all
 537 CDP messages generated by the device. Similarly, if there is
 538 a second address in the list, then it is considered to be the
 539 secondary management address of the device and should be the
 540 same in all CDP messages generated by the device.
 541
 542 Physical Location TLV: 0x0017
 543 This TLV, if present, contains a character string indicating
 544 the physical location of a connector which is on, or
 545 physically connected to, the interface over which the CDP
 546 packet containing this TLV is sent. This TLV's value will
 547 often be entered by the operator as a character string. If and
 548 when a more automated means of determining physical location is

available, then this TLV's value may have a specific structure.
The format of the TLV is:



Structured-Type - identifies the structure of the Structured-Value field, as follows:
 0 - an ASCII string
 1-255 - reserved for future use
 Structured-Value - the value formatted according to the structure given by the Structured-Type field. The length of this value is derived from the TLV's length.

External Port-ID TLV: 0x0018

Identifies the physical connector port on which the CDP packet is transmitted. This TLV is used in devices, such as those with optical ports, in which signals from multiple hardware interfaces are multiplexed through a single physical port. It contains the name of the external physical port through which the multiplexed signal is transmitted. External Port-ID TLV is encoded as an ASCII character string. The TLV length determines the length of the string. The value of this TLV must be the same as the MIB object ifName for the ifTable entry for the external port.

Procedure

All cisco devices send CDP packets periodically. They advertise a time-to-live in seconds which indicate the length of time after receipt at which a receiver must discard the information contained in the packet. The time-to-live value should always be larger than the periodic transmission timer. The periodic timers should be jittered to avoid synchronization effects. CDP packets are sent with a time-to-live that is non-zero after an interface is being enabled and a time-to-live of 0 immediately prior to an interface being idled down. This provides for quick state discovery.

All cisco devices will receive CDP packets and cache the information presented in the packet. The cached information is available at least to network management. cisco devices will never forward a CDP packet. If any information changes from the last received packet, the new information is cached and the older information is discarded even if its time-to-live has not yet expired.

Note that discarding old information when new is received means that it is not possible to "fragment" CDP packets. That is, a

transmitter cannot use multiple CDP packets to hold more TLVs than can fit into one.

At link up time, a cisco device should send three CDP packets at one second intervals. This protects against the delay caused by initial loss of packets when a link is restarted.

On most interface types, CDP should be enabled by default. If CDP is disabled, then received CDP packets should be discarded.

On VLAN interfaces for point-to-point media such as ISL, CDP packets are sent only on the lowest numbered VLAN (preferably, VLAN #1), but CDP packets should be received on all VLANs, even for VLANs which are disabled on the port.

CDPv1 and CDPv2

The only TLVs allowed in CDPv1 packets are 0x0001 through 0x0007, and the TLV 0x0007 is optional.

CDPv2 allows any TLVs, both those defined above and those not yet defined. TLVs 0x0001 through 0x0006 must be included in all CDPv2 packets; inclusion of other TLVs is optional.

A device should provide a configuration option for the highest version number, say version-H, of CDP packets it is to send; this defaults to the highest version number supported. When a CDP packet is received, the first processing of the packet is to discard it if its version number is higher than version-H. If a CDP packet with a lowered numbered version, say version-L, is received on interface-I, then the receiving device will subsequently send not only a version-H packet on interface-I, but will also send a version-L packet on that interface. This sending of version-L packets on interface-I continues until the TTLs of all version-L packets received on interface-I have expired.

If a cache entry already exists for the sending device/port of a newly-received CDP packet, the version number in the new packet is compared to the version number of the packet from which the cached entry was obtained. If the new packet has a higher version, then the cache entry is overwritten based on the new packet; if the new packet has a lower version, then the cache entry is left unchanged.

Implementation Considerations

Other protocols may wish to piggy-back their "hello" messages within CDP packets, using the Protocol-Hello TLV. To support such a capability, a CDP implementation might implement a "registry" type function, where such protocols ask the CDP code to notify them when neighbours supporting the particular protocol come and go, as indicated by the presence/absence of Protocol-Hello TLVs in recently received CDP packets.

Suggested Defaults

The periodic frequency for CDP transmissions is 60 seconds.
The advertised time-to-live is 180 seconds.

References

- [1] ISO, "Information Processing Systems - Telecommunications and Information Exchange between Systems - Protocol for Exchange of Inter-domain Routeing Information among Intermediate Systems to Support Forwarding of ISO 8473 PDUs", ISO/IEC DIS 10747, August 1992.
- [2] ISO TR 9577: 1991, Information Processing Systems - Telecommunications and Information Exchange between Systems - Protocol identification in the Network Layer.

Copyright: 1998,1999,2000,2001,2002 Cisco Systems, Inc.